Test your knowledge

1. What are the six combinations of access modifier keywords and what do they do?

- Public:
  - accessible from everywhere in a project without any restrictions
- Private:
  - accessible only inside a class or a structure
- Protected:
  - accessible inside the class and in all classes that derive from that class
- Internal:
  - the access is limited exclusively to classes defined within the current project assembly
- Protected Internal:
  - The protected internal access modifier is a combination of protected and internal. As a result, we can access the protected internal member only in the same assembly or in a derived class in other assemblies.
- Private Protected:
  - combination of the private and protected keywords, we can access members inside the containing class or in a class that derives from a containing class, but only in the same assembly(project)

2.What is the difference between the static, const, and readonly keywords when applied to a type member?

CONST: needs to have value at compile time and by default: static. can be accessed through the name of the class.

Readonly: very similar to const. Can be used on field, but not on local variables. It can either be initialized when declared or at the constructor of our object. This keyword ensures that

a variable instance or property of an object cannot be modified after initialization, and such attempts will result in an exception.

CONST vs readonly: For the readonly keyword, the latest value is known by the runtime. For the const keyword, the value must be known by compile time

Static is differenced from them as it has to link to the class.

3. What does a constructor do?

It was called during the creation of an object in order to initialize the fields of the object.

4. Why is the partial keyword useful?

It is possible to split the definition of a class, a struct, an interface or a method over two or more source files. Each source file contains a section of the type or method definition, and all parts are combined when the application is compiled.

It's desirable when working on large projects or on the automatically generated code.

5. What is a tuple?

introduced in .NET Framework 4.0. tuple is a data structure that contains a sequence of elements of different data types. It can be used where you want to have a data structure to hold an object with properties, but you don't want to create a separate type for it

6. What does the C# record keyword do?

7. What does overloading and overriding mean?

Overloading occurs when two or more methods in one class have the same method name but different parameters. Overriding occurs when two methods have the same method name and parameters. One of the methods is in the parent class, and the other is in the child class

8. What is the difference between a field and a property?

Properties expose fields. Fields should (almost always) be kept private to a class and accessed via get and set properties

9. How do you make a method parameter optional?

By using default value such as

```
public void my_add(string str1, string str2,
        string str3 = "GeeksforGeeks")
```

10. What is an interface and how is it different from abstract class?

1. Interface for interface and abstract class for abstract class in declaration
2. You can't have constructor in interface but you can have in abstract class

11. What accessibility level are members of an interface? public by default because the purpose of an interface is to enable other types to access a class or struct.

12. True/False. Polymorphism allows derived classes to provide different implementations of the same method.True

13. True/False. The override keyword is used to indicate that a method in a derived class is providing its own implementation of a method. True

14. True/False. The new keyword is used to indicate that a method in a derived class is providing its own implementation of a method. False
15. True/False. Abstract methods can be used in a normal (non-abstract) class. False
16. True/False. Normal (non-abstract) methods can be used in an abstract class. True
17. True/False.Derived classes can override methods that were virtual in the base class. True
18. True/False. Derived classes can override methods that were abstract in the base class.True
19. True/False. In a derived class, you can override a method that was neither virtual non abstract in the base class. False
20. True/False. A class that implements an interface does not have to provide an implementation for all of the members of the interface. False
21. True/False. A class that implements an interface is allowed to have other members that aren't defined in the interface. True
22. True/False. A class can have more than one base class. False
23. True/False. A class can implement more than one interface True